Image keyword tool

Release 0.0.1

Andrey Shpak

May 25, 2020

CONTENTS

1 Image Keyword Tool **3**
- 1.1 Features . . . 3
- 1.2 Installation . . . 3
- 1.3 Usage instructions . . . 3
- 1.4 Documentation . . . 4
- 1.5 Contributions . . . 4
- 1.6 Code of Conduct . . . 4
- 1.7 License . . . 4

2 Contributing **5**
- 2.1 Types of Contributions . . . 5
- 2.2 Setting Up the Code for Local Development . . . 6
- 2.3 Contributor Guidelines . . . 7
- 2.4 Testing with nox . . . 8
- 2.5 Becoming a Core Committer . . . 8

3 Contributors Code of Conduct **11**
- 3.1 Our Pledge . . . 11
- 3.2 Our Standards . . . 11
- 3.3 Our Responsibilities . . . 11
- 3.4 Scope . . . 12
- 3.5 Enforcement . . . 12
- 3.6 Attribution . . . 12

4 Changelog **13**
- 4.1 0.2.0 Quality Time (23-05-2020) . . . 13
- 4.2 0.1.0 Join the world (10-05-2020) . . . 13

5 Contributors list **15**
- 5.1 Development Leads . . . 15

Image Keyword Tool (IKT) is command-line utility that simplify usage of deep learning libraries for image tags generation.

CHAPTER

ONE

IMAGE KEYWORD TOOL

Image Keyword Tool (IKT) is command-line utility that simplify usage of deep learning libraries for image tags generation.

This tool is in active development. Most futures only in planned state now.

1.1 Features

- Python 3.7+ only
- Linux, Mac and Windows support
- [planned] JPG images recognition and tagging
- [planned] Convert PNG images to JPG images for tagging
- [planned] Write tags to JPGs exif
- [planned] Write tags to separate txt files
- [planned] Recognition threshold configuration
- [planned] Pluggable and selectable IA modules
- [planned] Pluggable modules for stock sites upload

1.2 Installation

```
pip install ikt
```

1.3 Usage instructions

```
ikt --help
```

1.4 Documentation

Check out the documentation for more info.

1.5 Contributions

Contributions are very welcome. Please check *contributors Guide*.

1.6 Code of Conduct

Everyone interacting in the Image Keywords tools project's codebases, issue trackers, chat rooms, and mailing lists is expected to follow the *Code of Conduct*.

1.7 License

CHAPTER

TWO

CONTRIBUTING

Contributions are welcome, and they are greatly appreciated! Every little bit helps, and credit will always be given.

- *Types of Contributions*
- *Contributor Setup*
- *Contributor Guidelines*
- *Contributor Testing*
- *Becoming a Core Committer*

2.1 Types of Contributions

You can contribute in many ways:

2.1.1 Report Bugs

Report bugs at Github Issue Tracker.

If you are reporting a bug, please include:

- Your operating system name and version.
- Any details about your local setup that might be helpful in troubleshooting.
- If you can, provide detailed steps to reproduce the bug.
- If you don't have steps to reproduce the bug, just note your observations in as much detail as you can. Questions to start a discussion about the issue are welcome.

2.1.2 Fix Bugs

Bugs are usually marked with 'bug' label.

- Look through the GitHub issues for bugs and issues.
- Any bug or problem issue without WIP label or related pull requests is open to whoever wants to implement it.
- If several fix approaches possible feel free to discuss possible fixes before actual codding.

2.1.3 Implement Features

Feature requests are usually marked with "enhancement" or "please-help" labels.

- Look through the GitHub issues for feature requests.
- Any feature request without WIP label or related pull requests is open to whoever wants to implement it.
- Please do not combine multiple feature enhancements into a single pull request.
- Feel free to create alternative features implementation if issue exist, but related pull request not yet merged. Please discuss in original pull request first.

2.1.4 Write Documentation

Image Keyword Tool (IKT) could always use more documentation, whether as part of the official IKT docs, in docstrings, or even on the web in blog posts, articles, and such.

If you want to review your changes on the documentation locally, after *Contributor Setup* you can do:

```
nox -s docs
```

This will compile the documentation, open it in your browser and start watching the files for changes, recompiling as you save.

2.1.5 Submit Feedback

The best way to send feedback is to file an issue at Github Issue Tracker.

If you are proposing a feature:

- Explain in detail how it would work.
- Keep the scope as narrow as possible, to make it easier to implement.
- Remember that this is a volunteer-driven project, and that contributions are welcome :)

2.2 Setting Up the Code for Local Development

Here's how to set up `ikt` for local development.

- Fork the `ikt` repo on GitHub.
- Clone your fork locally:

```
git clone git@github.com:your_name_here/ikt.git
```

- Install your local copy into a virtualenv. Assuming you have venv installed, this is how you set up your fork for local development:

```
pip install -e .
```

- Install development tools and dependencies into same virtualenv.

```
pip install -r requirements.txt
```

- Configure pre-commit hooks automation for future git commits. More info at pre-commit documentation

```
pre-commit install
```

- Create a branch for local development:

```
git checkout -b name-of-your-bugfix-or-feature
```

Now you can make your changes locally.

- When you're done making changes, check that your changes pass the tests and lint check:

```
nox
```

Please note that nox runs lint check automatically, since we have a test environment for it. Linting is done with pre-commit verification, so it will also correct some errors on this stage.

If you feel like running only the lint environment, please use the following command:

```
nox -s linting
```

- Ensure that your feature or commit is fully covered by tests. Check report after regular nox run.

You report will be placed to `htmlcov` directory. Please do not include this directory to your commits. By default this directory in our `.gitignore` file.

- Commit your changes and push your branch to GitHub:

```
git add .
git commit -m "Your detailed description of your changes."
git push origin name-of-your-bugfix-or-feature
```

- Submit a pull request through the GitHub website.

2.3 Contributor Guidelines

2.3.1 Pull Request Guidelines

Before you submit a pull request, check that it meets these guidelines:

1. The pull request should include tests.
2. If the pull request adds functionality, the docs should be updated. Put your new functionality into a function with a docstring, and add the feature to the list in README.md.
3. The pull request must pass all CI/CD jobs before being ready for review.
4. If one CI/CD job is failing for unrelated reasons you may want to create another PR to fix that first.

2.3.2 Coding Standards

- PEP8
- Black formatting
- Docstrings required
- Try to use type annotations, if possible
- Write all code in Python 3.7+.
- Use f-strings

2.3.3 Documentation Standards

- Markdown preferred over RST
- Line length: recommended 80 symbols, maximum 88, links are exceptions.
- Numbered lists preferred syntax use `1.` only.
- Unnumbered lists preferred syntax use `*`.

2.4 Testing with nox

To run all tests using various versions of python in virtualenv defined in `noxfile.py`, just run nox:

```
nox
```

This configuration file setup the pytest-cov plugin and it is an additional dependency. It generate a coverage report after the tests.

It is possible to tests with pre-selected version of python, to do this the command is:

```
nox -s tests-3.7 tests-3.8
```

Will run pytest with the python3.7 and python3.8 example.

Nox is new tool and in active development. Please check nox documentation for last updates.

To list all available nox sessions and their status you can use:

```
nox -l
```

docs session is disabled for automatic run and intended only for interactive launch for documentation development.

2.5 Becoming a Core Committer

Contributors may be given core commit privileges. Preference will be given to those with:

1. Past contributions to IKT and other open-source projects. Contributions to IKT include both code (both accepted and pending) and friendly participation in the issue tracker. Quantity and quality are considered.
2. A coding style that the other core committer find simple, minimal, and clean.
3. Access to resources for cross-platform development and testing.

4. Time to devote to the project regularly.

CHAPTER

THREE

CONTRIBUTORS CODE OF CONDUCT

3.1 Our Pledge

In the interest of fostering an open and welcoming environment, we as contributors and maintainers pledge to making participation in our project and our community a harassment-free experience for everyone, regardless of age, body size, disability, ethnicity, sex characteristics, gender identity and expression, level of experience, education, socio-economic status, nationality, personal appearance, race, religion, or sexual identity and orientation.

3.2 Our Standards

Examples of behavior that contributes to creating a positive environment include:

- Using welcoming and inclusive language
- Being respectful of differing viewpoints and experiences
- Gracefully accepting constructive criticism
- Focusing on what is best for the community
- Showing empathy towards other community members

Examples of unacceptable behavior by participants include:

- The use of sexualized language or imagery and unwelcome sexual attention or advances
- Trolling, insulting/derogatory comments, and personal or political attacks
- Public or private harassment
- Publishing others' private information, such as a physical or electronic address, without explicit permission
- Other conduct which could reasonably be considered inappropriate in a professional setting

3.3 Our Responsibilities

Project maintainers are responsible for clarifying the standards of acceptable behavior and are expected to take appropriate and fair corrective action in response to any instances of unacceptable behavior.

Project maintainers have the right and responsibility to remove, edit, or reject comments, commits, code, wiki edits, issues, and other contributions that are not aligned to this Code of Conduct, or to ban temporarily or permanently any contributor for other behaviors that they deem inappropriate, threatening, offensive, or harmful.

3.4 Scope

This Code of Conduct applies both within project spaces and in public spaces when an individual is representing the project or its community. Examples of representing a project or community include using an official project e-mail address, posting via an official social media account, or acting as an appointed representative at an online or offline event. Representation of a project may be further defined and clarified by project maintainers.

3.5 Enforcement

Instances of abusive, harassing, or otherwise unacceptable behavior may be reported by contacting the project team at cc@ashpak.ru. All complaints will be reviewed and investigated and will result in a response that is deemed necessary and appropriate to the circumstances. The project team is obligated to maintain confidentiality with regard to the reporter of an incident. Further details of specific enforcement policies may be posted separately.

Project maintainers who do not follow or enforce the Code of Conduct in good faith may face temporary or permanent repercussions as determined by other members of the project's leadership.

3.6 Attribution

This Code of Conduct is adapted from the Contributor Covenant, version 1.4, available at source link.

For answers to common questions about this code of conduct, see faq.

CHAPTER

FOUR

CHANGELOG

All notable changes to this project will be documented in this file.

The format is based on Keep a Changelog, and this project adheres to Semantic versioning.

Fresh changes should be at the beginning

4.1 0.2.0 Quality Time (23-05-2020)

This release is focused on further project CI/CD configuration. All related issues and pull requests can be found in the second closed project.

4.1.1 Added

- Documentation template for issues (#30) @insspb
- Codecov and codeclimate report upload added (#29) @insspb
- Added: Labeler workflow and labels configuration (#32) @insspb

4.1.2 Changed

- Release drafter configuration now uses dedicated 'log:' labels. (#28) @insspb

4.1.3 Removed

- Removed: scrutinizer support #33 (#34) @insspb

4.2 0.1.0 Join the world (10-05-2020)

This release is focused on project initial CI/CD configuration. All related issues and pull requests can be found in first closed project.

4.2.1 New features or options that was not exist before

- Added: Readme, Code of conduct and Contributing guide (#25) @insspb
- Added: Integration with pypi.org for auto deploy on new repo tag (#22) @insspb
- Added: Pytest tests and nox workflows, including linting (#21) @insspb
- Added: Pre-commit configuration and github actions workflow with black and other linting tools (#20) @insspb
- Added: Release drafter for automatically changelog generation (#18) @insspb
- Added: Issues and pull request templates (#17) @insspb
- Added: Empty package suitable for pip publishing without errors (#16) @insspb
- Added: License file #1 (#15) @insspb
- Added: Sphinx documentation build and integration with readthedocs (#13) @insspb

CHAPTER

FIVE

CONTRIBUTORS LIST

5.1 Development Leads

- Andrey Shpak (@insspb)